SESSION 5: Data Management Using R

<u>Assignment 3</u>

1. Test whether two vectors are exactly equal (element by element)

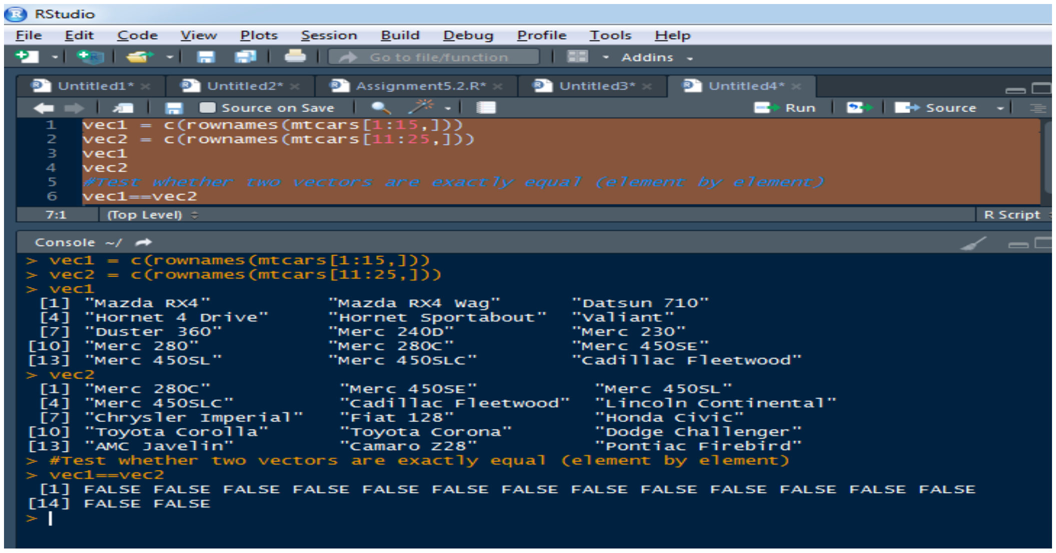
    vec1 = c(rownames(mtcars[1:15,]))

    vec2 = c(rownames(mtcars[11:25,]))

Answer:

vec1==vec2

Output:



```
vec1 = c(rownames(mtcars[1:15,]))
vec2 = c(rownames(mtcars[11:25,]))
vec1
vec2
#Test whether two vectors are exactly equal (element by element)
vec1==vec2
```

```
> vec1 = c(rownames(mtcars[1:15,]))
> vec2 = c(rownames(mtcars[11:25,]))
> vec1
 [1] "Mazda RX4"           "Mazda RX4 Wag"       "Datsun 710"
 [4] "Hornet 4 Drive"      "Hornet Sportabout"   "Valiant"
 [7] "Duster 360"          "Merc 240D"           "Merc 230"
[10] "Merc 280"            "Merc 280C"           "Merc 450SE"
[13] "Merc 450SL"          "Merc 450SLC"         "Cadillac Fleetwood"
> vec2
 [1] "Merc 280C"           "Merc 450SE"          "Merc 450SL"
 [4] "Merc 450SLC"         "Cadillac Fleetwood"  "Lincoln Continental"
 [7] "Chrysler Imperial"   "Fiat 128"            "Honda Civic"
[10] "Toyota Corolla"      "Toyota Corona"       "Dodge Challenger"
[13] "AMC Javelin"         "Camaro Z28"          "Pontiac Firebird"
> #Test whether two vectors are exactly equal (element by element)
> vec1==vec2
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[14] FALSE FALSE
> |
```

2. Sort the character vector in ascending order and descending order

    vec1 = c(rownames(mtcars[1:15,]))

    vec2 = c(rownames(mtcars[11:25,]))

Anwer:

#vec1 Assending and Desending

sort(vec1, decreasing=F)

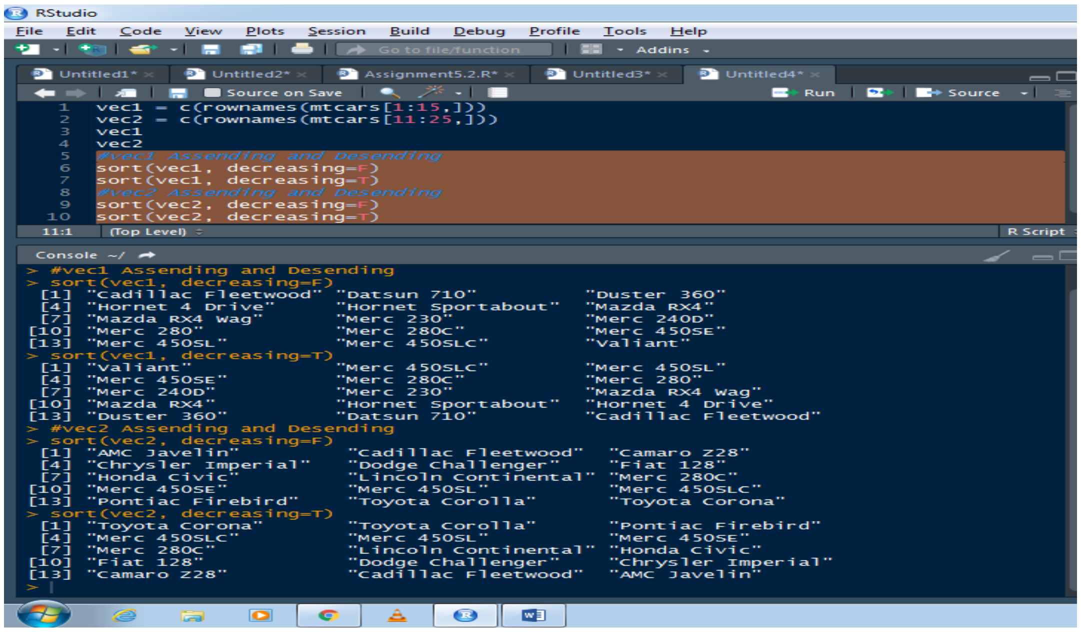
sort(vec1, decreasing=T)

#vec2 Assending and Desending

sort(vec2, decreasing=F)

sort(vec2, decreasing=T)

Output:



```
vec1 = c(rownames(mtcars[1:15,]))
vec2 = c(rownames(mtcars[11:25,]))
vec1
vec2
#vec1 Assending and Desending
sort(vec1, decreasing=F)
sort(vec1, decreasing=T)
#vec2 Assending and Desending
sort(vec2, decreasing=F)
sort(vec2, decreasing=T)
```

```
> #vec1 Assending and Desending
> sort(vec1, decreasing=F)
 [1] "Cadillac Fleetwood" "Datsun 710"          "Duster 360"
 [4] "Hornet 4 Drive"     "Hornet Sportabout"   "Mazda RX4"
 [7] "Mazda RX4 Wag"      "Merc 230"            "Merc 240D"
[10] "Merc 280"           "Merc 280C"           "Merc 450SE"
[13] "Merc 450SL"         "Merc 450SLC"         "Valiant"
> sort(vec1, decreasing=T)
 [1] "Valiant"            "Merc 450SLC"         "Merc 450SL"
 [4] "Merc 450SE"         "Merc 280C"           "Merc 280"
 [7] "Merc 240D"          "Merc 230"            "Mazda RX4 Wag"
[10] "Mazda RX4"          "Hornet Sportabout"   "Hornet 4 Drive"
[13] "Duster 360"         "Datsun 710"          "Cadillac Fleetwood"
> #vec2 Assending and Desending
> sort(vec2, decreasing=F)
 [1] "AMC Javelin"         "Cadillac Fleetwood"  "Camaro Z28"
 [4] "Chrysler Imperial"   "Dodge Challenger"    "Fiat 128"
 [7] "Honda Civic"         "Lincoln Continental" "Merc 280C"
[10] "Merc 450SE"          "Merc 450SL"          "Merc 450SLC"
[13] "Pontiac Firebird"    "Toyota Corolla"      "Toyota Corona"
> sort(vec2, decreasing=T)
 [1] "Toyota Corona"       "Toyota Corolla"      "Pontiac Firebird"
 [4] "Merc 450SLC"         "Merc 450SL"          "Merc 450SE"
 [7] "Merc 280C"           "Lincoln Continental" "Honda Civic"
[10] "Fiat 128"            "Dodge Challenger"    "Chrysler Imperial"
[13] "Camaro Z28"          "Cadillac Fleetwood"  "AMC Javelin"
> |
```

3. What is the major difference between str_c() and paste(). Show an example.

Answer:

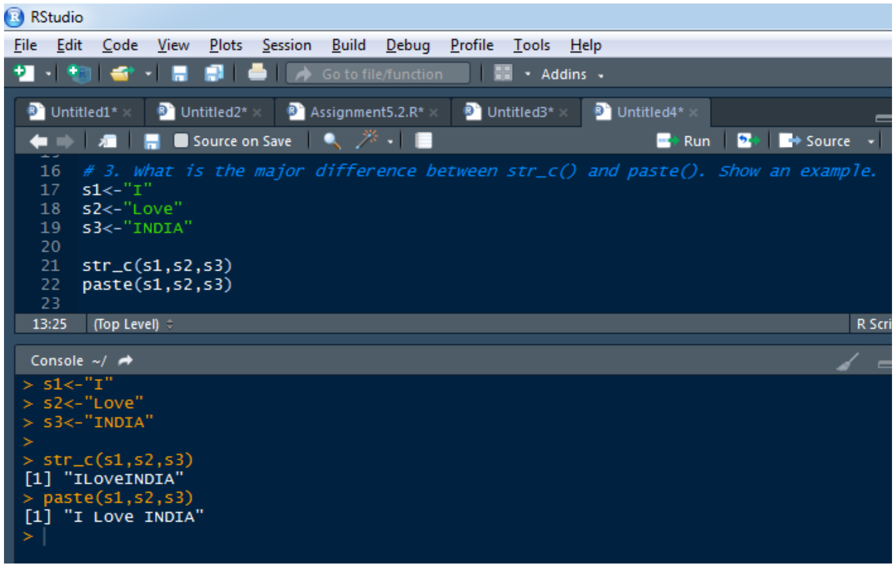
str_c() and **paste()** both function concatenate the strings, but **str_c()** don't have any separator in between the strings, while **paste()** by default will have space as the separator.

Below is the example:

s1<-"I"
s2<-"Love"
s3<-"INDIA"

str_c(s1,s2,s3)
paste(s1,s2,s3)

Output:



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1* Untitled2* Assignment5.2.R* Untitled3* Untitled4*
Source on Save
Run
Source
13:25
(Top Level)
R Scri
Console ~/

```
# 3. What is the major difference between str_c() and paste(). Show an example.
s1<-"I"
s2<-"Love"
s3<-"INDIA"

str_c(s1,s2,s3)
paste(s1,s2,s3)

```

```
> s1<-"I"
> s2<-"Love"
> s3<-"INDIA"
>
> str_c(s1,s2,s3)
[1] "ILoveINDIA"
> paste(s1,s2,s3)
[1] "I Love INDIA"
>
```

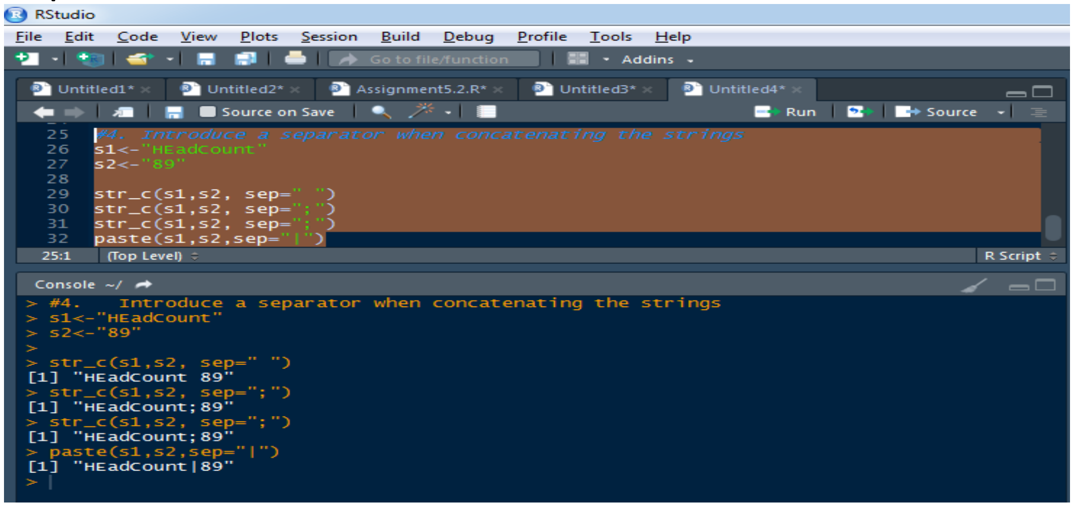
4. Introduce a separator when concatenating the strings

Answer:

s1<-"HEadCount"
s2<-"89"

str_c(s1,s2, sep=" ")
str_c(s1,s2, sep=";")
str_c(s1,s2, sep=";")
paste(s1,s2,sep="|")

Output:



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1* Untitled2* Assignment5.2.R* Untitled3* Untitled4*
Source on Save
Run
Source
25:1
(Top Level)
R Script
Console ~/

```
#4. Introduce a separator when concatenating the strings
s1<-"HEadCount"
s2<-"89"

str_c(s1,s2, sep=" ")
str_c(s1,s2, sep=";")
str_c(s1,s2, sep=";")
paste(s1,s2,sep="|")
```

```
> #4.   Introduce a separator when concatenating the strings
> s1<-"HEadCount"
> s2<-"89"
>
> str_c(s1,s2, sep=" ")
[1] "HEadCount 89"
> str_c(s1,s2, sep=";")
[1] "HEadCount;89"
> str_c(s1,s2, sep=";")
[1] "HEadCount;89"
> paste(s1,s2,sep="|")
[1] "HEadCount|89"
>
```